Using AI to generate traction maps and predict substrate stiffness for single cell traction force microscopy.

Sakib Mohammad[1], Fayed Alshammari[1], Arpan Roy[2], Sydney L Sarver[2], Farhan H Chowdhury[1, 2]

[1] *School of Electrical, Computer, and Biomedical Engineering, Southern Illinois University Carbondale, Carbondale, IL 62901, U.S.A.; sakib.mohammad@siu.edu (S.M.); fayed.alshammari@siu.edu (F.A.); farhan.chowdhury@siu.edu (F.C.)*

[2] *School of Mechanical, Aerospace, and Materials Engineering, Southern Illinois University Carbondale, Carbondale, IL 62901, U.S.A.; arpan.roy@siu.edu (A.R.); sydney.sarver@siu.edu (S.S.)*

Abstract

This project aims to predict single cell traction forces and substrate stiffness from phase image and/or bead displacement vectors, and cell area/shape. We cultured HeLa cells on substrate of two different stiffness and did the traction force microscopy (TFM) on them to generate the ground truth and used these data to train AI models. We used tree-based machine learning (ML) models and convolutional neural network (CNN) based deep learning (DL) models to obtain our goals. Both models were able to accurately predict the desired outcome. There were some shortcomings, however, owing to the tiny dataset and scarcity of computational resources to train complex DNN models. But the ideas can be extended to obtain meaningful results if sufficient resources are available.

Introduction

TFM is used to find the traction forces exhorted on the substrate a cell is attached, measure in Pascal (Pa)[1]. It is an important parameter for a cell that controls many cell behaviors and ultimately decides its fate[2]. Usually, special substrates of different Young's modulus are embedded with fluorescent beads and cells are seeded on them. Then the cells spread and displace the beads due to traction forces exhibited by the cells. Then the bead displacement is measured by comparing the extended state to the free state of the substrate when there are no cells. Traction is found out using a process called Fourier Transform Traction Cytometry (FTTC) where the bead displacement acts as the input[3]. The problem with that is one has to go through a lot of steps to obtain traction vectors of single cells using this process. If we could automate it somehow, or better yet, find the traction from only the phase image of the cells then it would save us a lot of effort. This is where Artificial Intelligence (AI) comes in.

AI has revolutionized the field of pattern recognition[4]. Each cell has a different morphology, and we can predict many properties including traction from these patterns. Dr. Yuli Wang, a researcher from Carnegie Mellon University proposed a DL based TFM method named DL-TFM back in 2021 that harnessed the ability of AI for pattern recognition to predict traction forces exhibited by cells[5]. However, his work had some shortcomings. In this work we want to address those shortcomings and extend his work using novel methods.

The contribution of this work is: (1) we used a tree based XGBoost[6] regression model to predict the traction vector fields from bead displacement vectors and shape of the cell (a parameter that Dr. Wang did not factor in their work) and showed it is possible to use a simple tree-based model to complete the task instead of going for a bulky DL model. (2) We predicted the stiffness of the substrate on which the cells were grown using another tree-based model CatBoost[7] classifier using the same parameters used before.

Since the displacement and shape of the cell are spatial quantity, an area where CNN excels in, (3) we used a 2D UNET[8] (as opposed to the 3D UNET used in Dr. Wang's work) model to predict the traction vectors from the quantities used in experiment (1). (4) Then we used an CNN (ResNet50[9]) to predict the stiffness of the substrate from only the phase image. (5) Next, we predicted the traction vectors from only phase image using the UNET. (6) Finally, we combined (3) and (5) and used both phase images and csv files (displacement vectors and cell shape vectors) to predict the traction. We will present the results in the following section.

Results

In table 1 we present the best results for predicting the traction vectors from bead displacement vectors and area vectors. All these vectors were 2 dimensional. We obtained the best results with an XGBoost regressor. For evaluation, we used typical metrics used for regression problems.

Table 1. Best regression scores using a tree-based model for predicting traction vectors.

Metric	Score
MSE	1.83e-21
MAE	3.64e-12
MAPE (%)	4.96

The results of our regression experiment seem promising and prove that we do not in fact need to use a UNET let alone a UNET that does 3D convolution to predict traction vectors which saves us a lot of time and computational resources and in addition we need a lot less data than a DL model to train a tree-based ML model. The following figure 1 shows the ground truth vs the predicted traction vector fields.

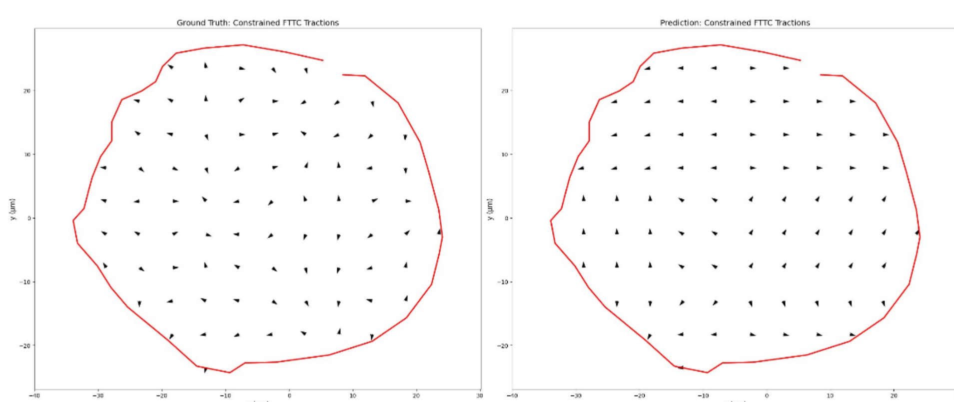

Figure 1. Ground truth vs the predicted traction (generated by XGBoost regressor) vector fields.

Then, in table 2 we can see the best results we obtained for predicting substrate stiffness from displacement vectors, traction vectors, and area/shape vectors using a CatBoost classifier. We used typical metrics used in binary classification to evaluate our model.

Table 2. Best classification scores using a tree-based model for predicting substrate stiffness.

Metric	Score
Accuracy (%)	77.95
Precision	0.82
Recall	0.74
F1 score	0.78

We can see that we obtained a respectable score in all categories from only 52 samples, however, there is much more room for improvement.

Next, we used a CNN to see if we could see an improvement in predicting the substrate stiffness from only the phase image of the cell. This will enable us to extract information about the substrate just from the phase image and we will not need to go into TFM analysis. The best results we got with a ResNet50, and it is presented in table 3.

Table 3. Best classification scores using a CNN for predicting substrate stiffness.

Metric	Score
Accuracy (%)	98
Precision	1.00
Recall	0.96
F1 score	0.98

From the table we can see we obtained a phenomenal result by using a CNN from only 52 images. So, we can conclude that this method is more robust than using a tree-based model that uses different TFM outputs to predict the substrate stiffness.

Now, let us come to predicting traction field from only phase image. This is a challenging task and we do not have a lot of data. We were able to train a UNET with a ResNet34[9] encoder to do the task and following is the best results obtained from a 3-fold cross validation. The results are presented below.

Table 4. Best regression scores using a CNN for predicting traction vectors from only the phase image.

Metric	Score
MAE	1.44e-10

MAPE (%)	37179.70

We can see the numbers are way off. But we can do better with a large dataset and more experimentation.

Since CNNs are good at analyzing spatial quantities, and the outputs of TFM analysis are all spatially dependent variables, we used a 2D CNN to predict the traction vectors using these variables. The following table shows the best results. We maintained similar experimental conditions as the previous experiment here.

Table 5. Best regression scores using a CNN for predicting traction vectors from TFM output variables.

Metric	Score
MAE	9.83e-12
MAPE (%)	17.87

We can see the results improve a lot and have more prospects than using only the phase images.

For the final experiment we combined the features extracted by two UNETs and made a combined prediction for traction vectors, i.e. combined the previous two experiments. Table 6 shows the results. This experiment was run for only an epoch to prove it was possible to do it this way.

Table 6. Best regression scores using two CNN for predicting traction vectors.

Metric	Score
MAE	2.00e-10
MAPE (%)	290.74

Discussion

For prediction of surface stiffness, we tried out several ML models, but we were relegated to using the tree-based model because there were row mismatches in our data and tree-based models can natively handle them without any modification to the data. We tried XGBoost, LightGBM[10] and CatBoost classifier and among them the last one worked the best for the classification task. For regression, however, only XGBoost regressor worked and neither LightGBM nor CatBoost regressor could handle the variability of rows in the data. In addition, since we did a multioutput regression, we could not use other linear ML models natively.

Our CNN experiment for predicting substrate stiffness was not very rigorous. We used only one CNN architecture; we did not even try out different hyperparameters. We did, however, used a 5-fold cross validation. If we employ a rigorous training regimen, we can make it even better.

The final 3 experiments using CNN does not paint the whole picture very well. We had a lack of data and computational resources. That is why we could not tune the hyperparameters properly. In addition, we need to verify it on other cell lines. We trained some of the models only for a few epochs just to prove our hypothesis and did not tune them further. However, for a full-blown work. We intend to address these issues.

Materials and Methods

Cell culture and image acquisition*:* HeLa cells (ATCC), a line of cervical cancer cells was regularly cultured in 10% Fetal Bovine Serum (FBS) (Gibco) supplemented Dulbecco's Modified Eagle Medium (DMEM) (Gibco). We made fluorescent bead (Sigma-Aldrich) embedded in Polyacrylamide (PAA) hydrogels with 0.6 kPa and 8.5 kPa stiffness on glass bottom dishes (Cellvis) using an in-house process. Then we trysinized (Gibco) the HeLa cells and seeded around 1000-3000 cell per dish. The next day, we took them for imaging with a Leica DMi 8 epifluorescence microscope. We took the phase image of the cells, then stretched gels with displaced beads images with a fluorescein isothiocyanate (FITC) filter. Then we trysinized the cells and took the relaxed gel images. We took these images and an in-house program to calculate bead displacement vectors, traction field vectors and shape of the cells using FTTC calculation and stored them in csv files. In total we had data for 52 single cells, among them we had an equal split for 26 cells belonging to 0.6 kPa stiffness and another 26 belonging to 8.5 kPa stiffness.

AI Experiments*:* Our AI experiments were divided into two categories. One is tree-based ML models, and the other is related to DL CNN models. We used XGBoost, CatBoost and LightGBM models (classifier and regressor) to predict gel stiffness and traction vector fields from the values obtained from TFM analysis. Since there were mismatches in the row numbers among the csv files, no regressor model except the XGBoost could handle it naturally. However, both XGBoost and CatBoost classifier could handle the classification task with satisfactory results. We employed grid-search to find out the optimum hyperparameters for the models and used 5 folds cross validation for a rigorous evaluation.

Then for the CNN based DL experiments we only used a ResNet50 for prediction of gel stiffness from cell phase images only. We used ImageNet[11] weights instead of random weight initialization and tuned from that. We employed a 5-fold cross validation to evaluate the model, however, we did not tune the hyperparameters as we just wanted to know if these experiments were possible or not rather than to obtain the best results, a trend we maintained throughout the DL experiments.

Next, we used a regular 2D UNET to predict traction forces from bead displacement vectors and cell shape vectors as they are spatial quantities, and so the prediction could be better by a CNN which is made for working with spatial data. Like before, we utilized transfer learning by using ImageNet weights with a ResNet34 encoder in the UNET and fined tuned the weights. Next, we predicted the same quantity, but this time used phase images instead of vectors. Finally, we combined the phase images and the vectors and used a multimodal UNET with ResNet34 encoders like before to predict the traction fields. We ran the experiments for a few epochs and did not fine tune the hyperparameters.

We used Google Colab to run our experiments in Python 3 environments. We used NumPy[12], Pandas[13], Scikit learn[14], PyTorch[15], Segmentation models[16], XGBoost, CatBoost, and LightGBM libraries to run our experiments. Since we used a free tier subscription of Google Colab, we could not run our DL experiments for long, one of the reasons why we could not tune our hyperparameters is because that would mean running our computation intensive models for a long time.

Acknowledgement

Members of mechano-genomic lab and NIH grant R15GM148440.

Reference

1 Zancla, A., Mozetic, P., Orsini, M., Forte, G. & Rainer, A. A primer to traction force microscopy. *J Biol Chem* **298**, 101867 (2022). https://doi.org:10.1016/j.jbc.2022.101867

2 Vining, K. H. & Mooney, D. J. Mechanical forces direct stem cell behaviour in development and regeneration. *Nat Rev Mol Cell Biol* **18**, 728-742 (2017). https://doi.org:10.1038/nrm.2017.108

3 Butler, J. P., Tolić-Nørrelykke, I. M., Fabry, B. & Fredberg, J. J. Traction fields, moments, and strain energy that cells exert on their surroundings. *American Journal of Physiology-Cell Physiology* **282**, C595-C605 (2002). https://doi.org:10.1152/ajpcell.00270.2001

4 Li, C., Li, X., Chen, M. & Sun, X. in *2023 IEEE 6th International Conference on Electronic Information and Communication Technology (ICEICT).* 557-562.

5 Wang, Y.-l. & Lin, Y.-C. Traction force microscopy by deep learning. *Biophysical Journal* **120**, 3079-3090 (2021). https://doi.org:https://doi.org/10.1016/j.bpj.2021.06.011

6 Chen, T. & Guestrin, C. (ACM).

7 Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V. & Gulin, A. in *Proceedings of the 32nd International Conference on Neural Information Processing Systems* 6639–6649 (Curran Associates Inc., Montréal, Canada, 2018).

8 Ronneberger, O., Fischer, P. & Brox, T. 234-241 (Springer International Publishing).

9 He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770-778 (2015).

10 Ke, G. *et al.* in *Proceedings of the 31st International Conference on Neural Information Processing Systems* 3149–3157 (Curran Associates Inc., Long Beach, California, USA, 2017).

11 Russakovsky, O. *et al.* ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision* **115**, 211-252 (2014).

12 Harris, C. R. *et al.* Array programming with NumPy. *Nature* **585**, 357-362 (2020). https://doi.org:10.1038/s41586-020-2649-2

13 McKinney, W.

14 Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).

15 Paszke, A. *et al.* in *Neural Information Processing Systems.*

16 Segmentation Models Pytorch (GitHub, GitHub repository, 2019).